CSCI 3104, Algorithms   **Profs. Grochow & Layer**
Problem Set 3   **Spring 2019, CU-Boulder**

Hyperlinks for convenience: 1a 1b 1c   2a 2b 2c   3a 3b 3c

1. *(10 pts total) For parts* (1a) *and* (1b), *justify your answers in terms of deterministic QuickSort, and for part* (1c), *refer to Randomized QuickSort. In both cases, refer to the versions of the algorithms given in the lecture notes for Week 3.*

   (a) *(3 points) What is the asymptotic running time of QuickSort when every element of the input A is identical, i.e., for $1 \leq i, j \leq n$, $A[i] = A[j]$? Prove your answer is correct.*

CSCI 3104, Algorithms   **Profs. Grochow & Layer**
Problem Set 3   **Spring 2019, CU-Boulder**

(b) *(3 points) Let the input array A be $[2, 1, -1, 4, 5, -4, 6, -3, 3, 0]$. What is the number of times a comparison is made to the element with value -3 (note the minus sign)?*

Name:

ID:

CSCI 3104, Algorithms   **Profs. Grochow & Layer**
Problem Set 3   **Spring 2019, CU-Boulder**

(c) *(4 points) How many calls are made to* `random-int` *in (i) the worst case and (ii) the best case? Give your answers in asymptotic notation.*

CSCI 3104, Algorithms   **Profs. Grochow & Layer**
Problem Set 3   **Spring 2019, CU-Boulder**

2. *(20 pts total) Professor Flitwick needs your help. He gives you an array A consisting of n integers $A[1], A[2], \ldots, A[n]$ and asks you to output a two-dimensional $n \times n$ array B in which $B[i, j]$ (for $i < j$) contains the sum of array elements $A[i]$ through $A[j]$, i.e., $B[i, j] = A[i] + A[i + 1] + \cdots + A[j]$. (The value of array element $B[i, j]$ is left unspecified whenever $i \geq j$, so it doesn't matter what the output is for these values.)*

   Flitwick suggests the following simple algorithm to solve this problem:

```
flitwickSolve(A) {
   for i = 1 to n {
      for j = i+1 to n {
         s = sum(A[i..j])        // look very closely here
         B[i,j] = s
}}}
```

   (a) *(5 points) Choose a function f such that $\Omega(f(n))$ serves as a bound on the running time of this algorithm, and prove that this is the case—that is, prove that the worst-case running time is at least $\Omega(f(n))$ on inputs of size n.*

Name: [ ]

ID: [ ]

CSCI 3104, Algorithms   **Profs. Grochow & Layer**
Problem Set 3   **Spring 2019, CU-Boulder**

(b) *(5 points) For this same function f you chose in part 2a, show that the running time of the algorithm on any input of size n is also $O(f(n))$. (This shows an asymptotically tight bound of $\Theta(f(n))$ on the running time.)*

Name:

ID:

CSCI 3104, Algorithms   Profs. Grochow & Layer
Problem Set 3   Spring 2019, CU-Boulder

(c) *(10 points) Although Flitwick's algorithm is a natural way to solve the problem—after all, it just iterates through the relevant elements of B, filling in a value for each—it contains some highly unnecessary sources of inefficiency. Give an algorithm that solves this problem in time $O(f(n)/n)$ (asymptotically faster than before) and prove its correctness. (Recall the class conventions on what is required when we ask you to "give an algorithm.")*

CSCI 3104, Algorithms   **Profs. Grochow & Layer**
Problem Set 3   **Spring 2019, CU-Boulder**

3. *(30 pts total) The Dementors have captured n prisoners, of dubious morals and dubious reliability. The Ministry of Magic needs your help to identify which prisoners can be relied on to tell the truth, and which cannot. To this end, they have developed a spell that can be cast on a pair of prisoners at a time. When the spell is cast, the two prisoners each say whether the other prisoner is truthful or a liar. A truthful prisoner always gives the correct answer—that is, correctly identifies whether the* other *prisoner is truthful or a liar—but anything one of the lying prisoners says cannot be trusted. That is, lying prisoners can lie whenever they want, but they* don't *necessarily always make false statements. Furthermore, the spell has the useful property that if it is cast on the same two prisoners repeatedly, they will always give the same answers they gave initially (hence why a spell is needed).*

   You quickly notice that there are four possible outcomes to the spell, when cast on prisoners i and j:

prisoner i says	prisoner j says		
"j is a liar"	"i is a liar"	$\implies$	at least one is a liar
"j is a liar"	"i is truthful"	$\implies$	at least one is a liar
"j is truthful"	"i is a liar"	$\implies$	at least one is a liar
"j is truthful"	"i is truthful"	$\implies$	both truthful, or both liars

   (a) *(10 points) Prove that if $n/2$ or more prisoners are liars, the Ministry cannot necessarily determine which prisoners are truthful using* any *strategy based on this kind of pairwise test. Assume a worst-case scenario in which the lying prisoners contain a psychic link that they can use to collectively conspire to fool the Ministry.*

CSCI 3104, Algorithms
Problem Set 3

Profs. Grochow & Layer
Spring 2019, CU-Boulder

(b) *(12 points) Suppose the Ministry knows that strictly more than $n/2$ of the prisoners are truthful, but not which ones. Prove that $\lfloor n/2 \rfloor$ pairwise tests are sufficient to reduce the problem to one of nearly half the size.*

CSCI 3104, Algorithms   **Profs. Grochow & Layer**
Problem Set 3   **Spring 2019, CU-Boulder**

(c) *(8 points) Now, under the same assumptions as part* (3b), *prove that all of the truthful prisoners can be identified with* $\Theta(n)$ *pairwise tests. Give and solve the recurrence (as always, in* O *or* Θ *notation) that describes the number of tests.*